Frontiers in Artificial Intelligence and Applications

ADVANCES IN KNOWLEDGE-BASED AND INTELLIGENT INFORMATION AND ENGINEERING SYSTEMS

Edited by
Marius Grana
Carlos Toro
Jorge Posada
Robert J. Howlett
Lakhmi C. Jain

Part 1

IOS Press

Advances in Knowledge-Based and Intelligent Information and Engineering Systems

Edited by

Manuel Graña
Computational Intelligence Group,
University of the Basque Country, UPV/EHU, Spain

Carlos Toro
Vicomtech-IK4, Spain

Jorge Posada
Vicomtech-IK4, Spain

Robert J. Howlett
Bournemouth University, United Kingdom

and

Lakhmi C. Jain
University South Australia, Australia

IOS
Press

Amsterdam • Berlin • Tokyo • Washington, DC

ISBN 978-1-61499-104-5 (print)
ISBN 978-1-61499-105-2 (online)
Library of Congress Control Number: 2012945596

Publisher
IOS Press BV
Nieuwe Hemweg 6B
1013 BG Amsterdam
Netherlands
fax: +31 20 687 0019
e-mail: order@iospress.nl

Distributor in the USA and Canada
IOS Press, Inc.
4502 Rachael Manor Drive
Fairfax, VA 22032
USA
fax: +1 703 323 3668
e-mail: iosbooks@iospress.com

Advances in Knowledge-Based and Intelligent Information and Engineering Systems
M. Graña et al. (Eds.)
IOS Press, 2012

doi:10.3233/978-1-61499-105-2-807

Networks of Polarized Evolutionary Processors as Problem Solvers*

Pedro Pablo Alarcón[1] **Fernando Arroyo**[2] **Victor Mitrana**[1]

[1]Department of Organization and Structure of Information,
University School of Informatics,
Polytechnic University of Madrid, Crta. de Valencia km. 7 - 28031 Madrid, Spain.
E-mail: pcavero@eui.upm.es, victor.mitrana@upm.es

[2]Department of Languages, Projects and Computer Information Systems,
University School of Informatics,
Polytechnic University of Madrid, Crta. de Valencia km. 7 - 28031 Madrid, Spain.
E-mail: farroyo@eui.upm.es

Abstract. In this paper, we propose a solution to an **NP**-complete problem, namely the "3-colorability problem", based on networks of polarized evolutionary processors. Our solution is uniform (it works for all instances of the same size) and time efficient (it works in linear time).

1 Introduction

Networks of evolutionary processors (NEP) form a class of highly parallel and distributed computing models inspired and abstracted from the biological evolution. On the other hand, NEPs resemble a pretty common architecture for parallel and distributed symbolic processing, related to the Connection Machine [3] as well as the Logic Flow paradigm [1]. A rather informal idea of what a network of evolutionary processor is consists of a virtual (complete) graph in which each node hosts a very simple processor called an evolutionary processor. By an evolutionary processor we mean a processor which is able to perform very simple operations, namely point mutations in a DNA sequence (insertion, deletion or substitution of a pair of nucleotides). By an informal parallelism with the natural process of evolution, each node may be viewed as a cell having genetic information encoded in DNA sequences which may evolve by local evolutionary events, that is point mutations. Each node processor, which is specialized just for one of these evolutionary operations, acts on the local data and then local data becomes a mobile agent which can navigate in the network following a given protocol. Only that data which is able to pass a filtering process can be communicated. This filtering process may require to satisfy some conditions imposed by the sending processor, by the receiving processor or by both of them. All the nodes send simultaneously their data and the receiving nodes handle also simultaneously all the arriving messages, according to some strategies. The

* Work supported by the Project TIN2011-28260-C03-03.

reader interested in a survey of the main results regarding NEPs is referred to [4].

This work proposes a new variant of NEP and discusses the potential of this variant for solving hard computational problems. The main and completely new feature of this variant is the valuation mapping which assigns to each string an integer value, depending on the values assigned to its symbols. Actually, we are not interested in computing the exact value of a string, but just the sign of this value. By means of this valuation, one may say that the strings are electrically polarized. Thus, if the nodes are polarized as well, the strings migration from one node to another through the channel between the two cells seems to be more natural.

We consider here networks of polarized evolutionary processors (NPEP) as problem solvers. When discussing the potential of a given NPEP to solve a problem one may say that one designs a NPEP-algorithm. We propose a NPEP-algorithm for a well-known **NP**-complete problem, namely the "3-colorability" problem, that requires a linear time in the size of the given instances.

2 Preliminaries

We start by summarizing the notions used throughout this work. An *alphabet* is a finite and nonempty set of symbols. The cardinality of a finite set A is written $card(A)$. Any finite sequence of symbols from an alphabet V is called *string* over V. The set of all strings over V is denoted by V^* and the empty string is denoted by ε. The length of a string x is denoted by $|x|$ while $alph(x)$ denotes the minimal alphabet W such that $x \in W^*$.

A homomorphism from the monoid V^* into the monoid (group) of additive integers $\mathbf{Z}$ is called *valuation* of V^* in $\mathbf{Z}$.

We say that a rule $a \to b$, with $a, b \in V \cup \{\varepsilon\}$ and $ab \neq \varepsilon$ is a *substitution rule* if both a and b are not ε; it is a *deletion rule* if $a \neq \varepsilon$ and $b = \varepsilon$; it is an *insertion rule* if $a = \varepsilon$ and $b \neq \varepsilon$. The set of all substitution, deletion, and insertion rules over an alphabet V are denoted by Sub_V, Del_V, and Ins_V, respectively. Given a rule σ as above and a string $w \in V^*$, we define the following *actions* of σ on w:

- If $\sigma \equiv a \to b \in Sub_V$, then $\sigma(w) = \begin{cases} \{ubv : \exists u, v \in V^* \ (w = uav)\}, \\ \{w\}, \text{ otherwise.} \end{cases}$
  A substitution rule $a \to b$ is applied to a string (one of its copies) in the usual way (one occurrence of a, nondeterministically chosen, is replaced by b). The same rule may replace different occurrences of the same symbol in different copies of the same string. Therefore, we implicitly assume that each string appears in an arbitrarily large number of copies.
- If $\sigma \equiv a \to \varepsilon \in Del_V$, then $\sigma(w) = \begin{cases} \{uv : \exists u, v \in V^* \ (w = uav)\}, \\ \{w\}, \text{ otherwise} \end{cases}$
- If $\sigma \equiv \varepsilon \to a \in Ins_V$, then $\sigma(w) = \{uav : \exists u, v \in V^* \ (w = uv)\}$.

For every rule σ and $L \subseteq V^*$, we define the action of σ on L by $\sigma(L) = \bigcup_{w \in L} \sigma(w)$. Given a finite set of rules M, we define the action of M on the

string w and the language L by:

$$M(w) = \bigcup_{\sigma \in M} \sigma(w) \quad \text{and} \quad M(L) = \bigcup_{w \in L} M(w),$$

respectively.

Definition 1. A *polarized evolutionary processor* over V is a pair (M, α), where:

- M is a set of substitution, deletion or insertion rules over the alphabet V. Formally: $(M \subseteq Sub_V)$ or $(M \subseteq Del_V)$ or $(M \subseteq Ins_V)$. The set M represents the set of evolutionary rules of the processor. As one can see, a processor is "specialized" in one evolutionary operation, only.
- $\alpha \in \{-, 0, +\}$ is the polarization of the node (negatively, neutral or positively charged, respectively).

We denote the set of evolutionary processors over V by EP_V. Clearly, the evolutionary processor described here is a mathematical concept similar to that of an evolutionary algorithm, both being inspired from the Darwinian evolution. As we mentioned in the Introduction, the rewriting operations we have considered might be interpreted as mutations and the filtering process described above might be viewed as a selection process. Recombination is missing but it was asserted that evolutionary and functional relationships between genes can be captured by taking only local mutations into consideration [5].

Definition 2. A *network of polarized evolutionary processors* (NPEP for short) is a 7-tuple $\Gamma = (V, U, G, \mathcal{R}, \varphi, \underline{In}, \underline{Out})$, where:

- $\diamond$ V and U are the input and network alphabet, respectively, $V \subseteq U$.
- $\diamond$ $G = (X_G, E_G)$ is an undirected graph without loops with the set of vertices X_G and the set of edges E_G. G is called the *underlying graph* of the network.
- $\diamond$ $\mathcal{R} : X_G \longrightarrow EP_U$ is a mapping which associates with each node $x \in X_G$ the polarized evolutionary processor $\mathcal{R}(x) = (M_x, \alpha_x)$.
- $\diamond$ φ is a valuation of U^* in $\mathbf{Z}$.
- $\diamond$ $\underline{In}, \underline{Out}, \in X_G$ are the *input* and the *output* node of Γ, respectively.

We say that $card(X_G)$ is the size of Γ. A *configuration* of a NPEP Γ as above is a mapping $C : X_G \longrightarrow 2^{V^*}$ which associates a set of strings with every node of the graph. A configuration may be understood as the sets of strings which are present in any node at a given moment. Given a string $w \in V^*$, the initial configuration of Γ on w is defined by $C_0^{(w)}(x_I) = \{w\}$ and $C_0^{(w)}(x) = \emptyset$ for all $x \in X_G \setminus \{x_I\}$.

A configuration can change either by an *evolutionary step* or by a *communication step*. When changing by an evolutionary step, each component $C(x)$ of the configuration C is changed in accordance with the set of evolutionary rules M_x associated with the node x. Formally, we say that the configuration C' is obtained in *one evolutionary step* from the configuration C, written as $C \Longrightarrow C'$, iff

$$C'(x) = M_x(C(x)) \text{ for all } x \in X_G.$$

When changing by a communication step, each node processor $x \in X_G$ sends out one copy of each string it has, which has a polarity different than that of x, to all the node processors connected to x and receives all the strings sent by any node processor connected with x providing that they have the same polarity as that of x. Note that, for simplicity reasons, we prefer to consider that a string migrate to a node with the same polarity and not an opposed one. Formally, we say that the configuration C' is obtained in *one communication step* from configuration C, written as $C \vdash C'$, iff

$$C'(x) = (C(x) \setminus \{w \in C(x) \mid \varphi(w) \not= \alpha_x\}) \cup \bigcup_{\{x,y\} \in E_G} (\{w \in C(y) \mid \alpha_y \not= \varphi(w) = \alpha_x\}),$$

for all $x \in X_G$. Note that strings that have the same polarity as that of the node in which they are remain in that node and can be further modified in the subsequent evolutionary steps, while strings with a different polarity are expelled. Further, each expelled string from a node x that cannot enter any node connected to x (no such node has the same polarity as the string has) is lost.

Let Γ be a NPEP, the computation of Γ on the input string $w \in V^*$ is a sequence of configurations $C_0^{(w)}, C_1^{(w)}, C_2^{(w)}, \ldots$, where $C_0^{(w)}$ is the initial configuration of Γ on w, $C_{2i}^{(w)} \Longrightarrow C_{2i+1}^{(w)}$ and $C_{2i+1}^{(w)} \vdash C_{2i+2}^{(w)}$, for all $i \geq 0$. Note that the configurations are changed by alternative steps. By the previous definitions, each configuration $C_i^{(w)}$ is uniquely determined by the configuration $C_{i-1}^{(w)}$. Otherwise stated, each computation in a NPEP is deterministic.

A computation as above *halts*, if one of the following two conditions is satisfied:

- There exists a configuration in which the set of strings existing in the output node $\underline{Out}$ is non-empty.
- No further step is possible.

Given a NPEP Γ and an input string w, we say that Γ accepts w if the computation of Γ on w halts with a non-empty output node. If Γ halts on every input, we say that Γ decides w.

Let Γ be a NPEP with the input alphabet V that halts on every input. The *time complexity* of the finite computation $C_0^{(x)}, C_1^{(x)}, C_2^{(x)}, \ldots C_m^{(x)}$ of Γ on $x \in V^*$ is denoted by $Time_\Gamma(x)$ and equals m.

The time complexity of Γ is the partial function from $\mathbf{N}$ to $\mathbf{N}$,

$$Time_\Gamma(n) = \max\{Time_\Gamma(x) \mid x \in V^*, |x| = n\}.$$

3 Solving Problems With NPEPs

We discuss briefly and informally how NPEPs could be used as problem solvers. A possible correspondence between decision problems and languages can be done

via an encoding function which transforms an instance of a given decision problem into a string, see, e.g., [2]. We say that a decision problem P is solved in time $O(f(n))$ by NPEPs if there exists a family $\mathcal{G}$ of NPEPs such that the following conditions are satisfied:

1. The encoding function of any instance p of P having size n can be computed by a deterministic Turing machine in time $O(f(n))$.
2. For each instance p of size n of the problem one can effectively construct, in time $O(f(n))$, a NPEP $\Gamma(p) \in \mathcal{G}$ which decides, again in time $O(f(n))$, the string encoding the given instance. This means that the string is decided if and only if the solution to the given instance of the problem is "YES". This effective construction is called an $O(f(n))$ time solution to the considered problem.

If a NPEP $\Gamma \in \mathcal{G}$ constructed above decides the language of strings encoding all instances of the same size n, then the construction of Γ is called a uniform solution. Intuitively, a solution is uniform if for problem size n, we can construct a unique NPEP solving all instances of size n taking the (reasonable) encoding of instance as "input".

In the followings we attack a decision problem known to be **NP**-complete and construct a uniform solution based on NPEP. The problem we deal with is the well known "3-colorability problem". This problem is to decide whether each vertex in a connected undirected graph can be colored by using three colors (say red, blue, and green) in such a way that after coloring, no two vertices which are connected by an edge have the same color. Graph coloring (we consider here vertex coloring) has numerous applications: scheduling, register allocation, pattern matching, frequency assignment for mobile radio stations, etc.

Theorem 1 *The "3-colorability problem" of a graph of order n and size m can be uniformly solved by NPEPs in $\mathcal{O}(n+m)$ time.*

Proof. Let $X = (T, Q)$ be a graph with $T = \{v_1, v_2, \ldots, v_n\}$ and $Q = \{e_1, e_2, \ldots, e_m\}$, where each e_k is given in the form $e_k = \{v_i, v_j\}$, for some $1 \leq i \neq j \leq n$. We construct the NPEP (following Definition 2) that will decide whether the graph X can be colored with three colors, namely *red, blue, green.*

$$\Gamma = (V, U, G, \mathcal{R}, \varphi, \underline{In}, \underline{Out})$$

with

$$\begin{aligned} V &= \{T_0, T_1, T_2, \ldots, T_{n(n-1)/2}\} \cup \{e_1, e_2, \ldots, e_{n(n-1)/2}\} \cup \{a\}, \\ U &= V \cup \{T'_0, T'_1, T'_2, \ldots, T'_{n(n-1)/2}\} \cup \{e'_1, e'_2, \ldots, e'_{n(n-1)/2}\} \cup \\ &\quad \{\bar{e}_1, \bar{e}_2, \ldots, \bar{e}_{n(n-1)/2}\} \cup \{r_i, b_i, g_i \mid 1 \leq i \leq n\} \cup \{T''_0, F\}. \end{aligned}$$

We now define the valuation mapping φ:

$$\begin{array}{ll}
\varphi(T_k) = 0, 0 \leq k \leq n(n-1)/2, & \varphi(T'_j) = 1, 0 \leq j \leq n(n-1)/2, \\
\varphi(z_i) = 0, 1 \leq i \leq n, z \in \{r, b, g\}, & \varphi(e_j) = 0, 1 \leq j \leq n(n-1)/2, \\
\varphi(z'_i) = 1, 1 \leq i \leq n, z \in \{r, b, g\}, & \varphi(e'_j) = -2, 1 \leq j \leq n(n-1)/2, \\
\varphi(z''_i) = 3, 1 \leq i \leq n, z \in \{r, b, g\}, & \varphi(\bar{e}_j) = 0, 1 \leq j \leq n(n-1)/2, \\
\varphi(a) = 1, & \varphi(F) = 1, \\
\varphi(T''_0) = -1. &
\end{array}$$

We first give the shape of the underlying graph G in Figure 1. Each box labeled by $[i, j]$, $1 \leq i < j \leq n$, encapsulates a subgraph associated with the edge $\{i, j\}$ which is described in Figure 2.

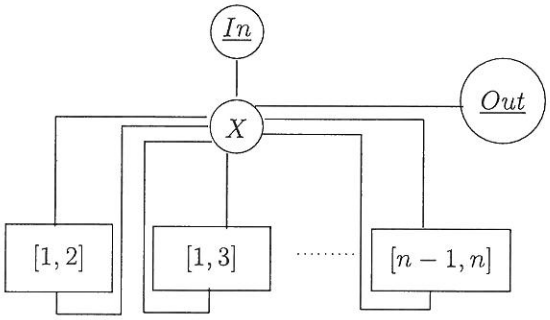

Figure 1. General shape of the underlying graph G

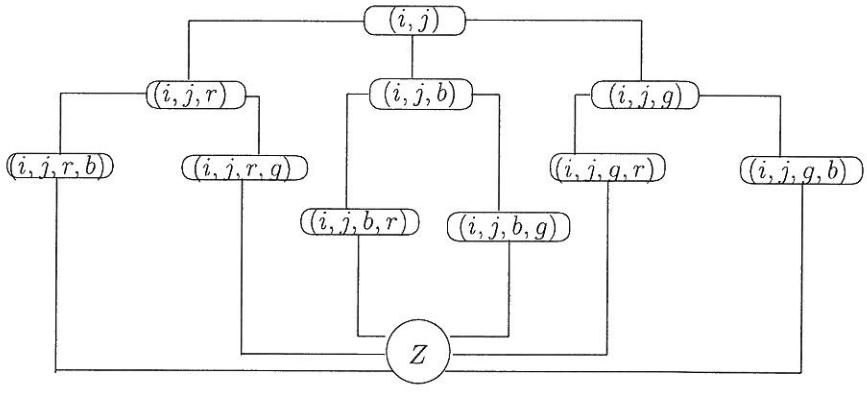

Figure 2. The subgraph associated with each edge $\{i, j\}$.

We now define the nodes of our network in Table 1:

Node	M	α
$\underline{In}$	$\bigcup_{i=1}^{n}\{a \to r_i, a \to b_i, a \to g_i\}$	+
X	$\{T_k \to T'_{k-1} \mid 1 \leq k \leq n(n-1)/2\} \cup \{T_0 \to T''_0\}$	0
(i,j)	$\{e_k \to e'_k \mid e_k = \{i,j\}\}$	+
$(i,j,z), z \in \{r,b,g\}$	$\{z_i \to z'_i\}$	−
$(i,j,z,y), z \in \{r,b,g\}$, $y \in \{r,b,g\}, z \neq y$	$\{y_j \to y''_j\}$	0
Z	$\{T'_k \to T_k \mid 0 \leq k \leq n(n-1)/2\} \cup$ $\{z'_i \to z_i, g''_j \to g_j\} \cup \{e'_k \to \bar{e}_k \mid e_k = \{i,j\}$	+
$\underline{Out}$	$\emptyset$	−

Table 1. The parameters of the nodes of Γ

We now discuss how the network accept any input string encoding a graph that can be colored with three colors and reject the input string if it encodes an invalid graph. The computation starts with a string of the form

$$w = T_m e_1 e_2 \dots e_m a^n,$$

which encodes a graph with n nodes (the number of a's equals the number of nodes) and m edges (the edges are encodes by the symbols e_k). Note that $\varphi(w)$ is positive. In the input one $\underline{In}$ the input word stays until all occurrences of a are replaced by r_i, b_i, g_i, for some $1 \leq i \leq n$. Indeed, as long as there are still occurrences of a in a string in $\underline{In}$, the value of that string is still positive. The meaning of replacing an occurrence of a by r_i is that the node v_i is colored in red. Therefore, after replacing all occurrences of a the input node contains strings encoding all possible colorings of the graph. Note that if a string contains both r_i and b_i, for some $1 \leq i \leq n$ (a node is colored by two colors in the same coloring), this means that a node is missing from this coloring and the string will be blocked during the computation.

Let us follows with an arbitrary string that migrates from $\underline{In}$ to the node X. Here T_m is replaced by T'_{m-1} and one copy of the string migrates to the input node, where it will remain forever, while $n(n-1)/2$ copies will migrate to the nodes (i,j), $1 \leq i < j \leq n$. As soon as a string arrives in (i,j), we distinguish two situations that may appear depending on whether the string contains a symbol e_k that encodes the edge $\{i,j\} \in Q$. If the string does not contains such a symbol, then it will remain blocked in this node. If such a symbol appears in the string, then e_k is replaced by its primed copy, the value of the new string becomes negative, and three copies of the new string simultaneously migrate to the nodes (i,j,r), (i,j,b), and (i,j,g), respectively. We continue our analysis with the string arriving in (i,j,r). An analogous discussion works for the other nodes as well. In (i,j,r) one checks whether or not the color assigned to the node v_i is red. This is done by replacing r_i with its primed copy. If the symbol r_i does not appear in the string, it is blocked in (i,j,r) for the rest of the computation. After having r_i substituted by r'_i, the new string gets a neutral value and two copies migrate to (i,j,r,b) and (i,j,r,g), respectively. In these nodes one checks

in a similar manner to that in (i, j, r) whether the color assigned to v_j is blue or green, respectively. Assume that the color assigned to v_j is actually blue. Then after replacing b_j with b_j'' the string enters node Z. Here all the primed symbols are restored except for e_k' which is replaced by $\bar{e}_k$. Only after replacing all there symbols, the new string returns to X and the whole process described above resumes. It is worth mentioning that the string arrives in X if the colors assigned to the nodes i and j in the coloring encoded by the string is valid. Now the coloring has passed successfully the test for the edge $\{i, j\}$ and the index of T has decreased by 1 which means that there are $m - 1$ edges for which the coloring is to be further tested.

As we have seen, all strings encoding invalid colorings remain blocked and only those encoding valid colorings return to X after each edge test. If the index of the symbol T becomes 0 in a given string, this means that the coloring encoded by that string has passed the tests for all edges of Q, therefore the input graph can be colored with three colors. The symbol T_0 in such a string is replaced by T_0'' in X which makes it to have a negative value. Consequently, it migrates to $\underline{Out}$ and the computation halts and the input string is accepted.

We mention that our solution is uniform because the network remains unchanged for any instance of a graph of order n. Furthermore, the total number of evolutionary steps is $n + 8m + 1$, given by:

- n steps in the node $\underline{In}$,
- m times the following steps:
  - one step in X,
  - one step in each node situated on every level in the subgraph associated with an edge,
  - four steps in Z.
- one last step in X.

In conclusion the NPEP constructed above uniformly solves the problem in $\mathcal{O}(n + m)$ time. □

4 Final Remarks

We have presented a uniform solution to the "3-colorability problem" in $\mathcal{O}(n+m)$ time based on NPEP. This results suggests that this computational model might have an interesting computational power which deserves to be investigated. We shall return to this topic in a forthcoming work.

References

1. Errico, L., Jesshope, C. (1994). Towards a new architecture for symbolic processing, In *Artificial Intelligence and Information-Control Systems of Robots '94*, 31–40.
2. Garey, M., Johnson, D. (1979). *Computers and Intractability. A Guide to the Theory of NP-completeness.* Freeman, San Francisco, CA.
3. Hillis, W.D. (1979). *The Connection Machine.* MIT Press, Cambridge.

4. F. Manea, C. Martín-Vide, V. Mitrana. Accepting networks of evolutionary word and picture processors: A survey. In *Scientific Applications of Language Methods*, Mathematics, Computing, Language, and Life: Frontiers in Mathematical Linguistics and Language Theory - Vol. 2, World Scientific 523–560, 2010.
5. D. Sankoff et al., Gene order comparisons for phylogenetic inference:evolution of the mitochondrial genome, Proceedings of the National Academy of Sciences of the United States of America **89**, (1992), 6575–6579.